**CPSC 1020 SPRING 2016**

**EXAM #1**

**This is a Closed Book exam. Please keep your notes and your computers closed. You have 50 minutes to complete the Exam. There is a total of 111 points (11 extra credit)**

**Question 1: (10 points)**

Consider the following program. What would the output be?

#include <stdio.h>

int main() {

unsigned int a = 60; /\* 60 = 0011 1100 \*/

unsigned int b = 13; /\* 13 = 0000 1101 \*/

int c = 0;

c = a & b;

printf("Line 1 - Value of c is %d\n", c );

c = a | b;

printf("Line 2 - Value of c is %d\n", c );

c = a ^ b;

printf("Line 3 - Value of c is %d\n", c );

c = a >> 2;

printf("Line 4 - Value of c is %d\n", c );

return 0;

}

Output:

**Question 2: (10 points):**

In class I discussed several reasons you would want to use pointers. **In no more than 2 sentences each, describe 2 of these reasons. You do not have to use the entire page for your answers.**

1.

2.

**Question 3: (10 points)**

Consider the following code.

#include <stdio.h>

int main()

{

int a = 112;

int b = -1;

float c = 3.14;

int \*d = &a;

float \*e = &c;

char arr[] = "Yvon";

//int \*ptr;

//\*ptr = 12;

printf("%i, %i, %f, %f, %s\n", a, \*d, c, \*e, arr);

return 0;

}

You have two task:

1. The two lines in the above code that have been commented out can (and did) cause this program to have a segfault, bus error, or memory fault error (uncommented). Briefly, explain why this code could cause this problem.

2. In the box below write the output of this program.

Output:

**Question 4: (15 points)**

Consider the following program. In the box below show the output of this program.

#include <stdio.h>

int main()

{

int values[10] = {-1, 14, -24, 6, 9, 2, -3, 4, 7, 3};

char word[26] = {'H', 'a', 'v', 'e', ' ', 'a', ' ','g', 'r', 'e',

'a', 't', ' ', 'w', 'e', 'e', 'k', 'e', 'n', 'd', '!', '!', '!', '!'};

int i, n=10;

int \*pv0 = values;

printf("0. pv0 = %i\n", \*pv0);

// output

printf("\n1. Check: ");

char \*pv1 = word;

for (i=0; i<5; i++)

{

printf("%c", \*pv1);

pv1 += 3;

}

return(0);

}

Output:

**Question 5: (10 points)**

Consider the following snippet of code and answer the question.

int num = 5;

int num2 ;

This is the “C” version of asking and receiving user input.

printf(“Please enter an integer between 1 and 100”);

scanf(“%d”, &num2);

In the box below write the C++ version of asking for and receiving user input.

C++ code goes here:

**Question 6: (5 points)**

Consider the following snippet of code:

int val;

double num = 3.7;

C-style casting looks like the following:

val = (int) num;

In the box below rewrite the C-style of casting using the post-standard C++ style of type casting.

**Question 7: (5 points)**

Both a C++ Constant Variable and a C Style #define are read only. They both must also be given a value when declared. In class we discussed a major difference between a C++ Constant Variable and a C Style of #define. **In no more than 2 sentences discuss this difference.**

1.

**Question 8: (5 points)**

C++ provides programmers with several formatting output stream manipulators. We discussed several of these in class. Match the following manipulator with the correct description.

\_\_\_\_\_\_\_\_\_\_\_ setw(n)

\_\_\_\_\_\_\_\_\_\_\_ fixed

\_\_\_\_\_\_\_\_\_\_\_ showpoint

\_\_\_\_\_\_\_\_\_\_\_ setprecision(n)

\_\_\_\_\_\_\_\_\_\_\_ left or right

1. Causes a decimal point and trailing zeros to be displayed for floating-point numbers, even if there is no fractional part.
2. Cause the output to all go to one side or another.
3. Sets the size of the print field.
4. Forces a floating-point number to display in a particular point format.
5. Allows us to set the number of significant digits to be printed. This includes numbers before and after the decimal.

**Question 9: (6 points)**

Given the following snippet of code, answer the questions:

double x = 456.0;

cout << x << endl;

What will print? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

cout << fixed << showpoint << setprecision(2) << x << endl;

What will print? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 10: (15 points)**

1.

In “C” we use FILE \*fp = fopen(“file.txt”, “r”) to open a file to read. (input)

Write the necessary code to open an input file for reading in C++.

2.

In “C” we use FILE \* fp = fopen(“file.txt”, “w”) to open a file to write to. (output)

Write the necessary code to open an output file for writing in C++.

3.

In “C” we use fclose(fp) to close a file that was previously opened.

Write the necessary code to close a file that has been opened for:

1. reading (input)

1. writing (output)

**Question 11: (20 points)**

**Fill in the specified lines of code below.**

#include <stdio.h>

#include <stdlib.h>

typedef struct{

char first\_name[15];

char last\_name[15];

int age;

}Student;

int main(int argc, char \*argv[]){

Student \*sptr;

int num\_students, i;

FILE \*fptr = fopen(argv[1], "r");

/\*This reads in the number of students that the file contains data for.\*/

fscanf(fptr, "%d", &num\_students);

**/\*FILL IN CODE HERE:**

**\*Malloc memory for the student information that will be read in\*/**

for(i = 0; i < num\_students; i++){

**/\*FILL IN CODE HERE: Using fscanf and printf, scan in the student information,**

**\*then print the information back out to the terminal.\*/**

}

printf("\n\n");

**/\*FILL IN CODE HERE: We are now finished with the malloced (dynamically**

**\*allocated memory) Write the line of code that returns the memory back to the**

**\*operating system. \*/**

fclose(fptr);

return 0;

}